Python: module cdms.selectors

cdms.selectors

index

Classes to support easy selection of climate data

Modules

MA   cdtime   string   types

Classes

- cdms.error.CDMSError(exceptions.Exception)
    - SelectorError
- Selector
- SelectorComponent
    - axisComponent
        - coordinateComponent
        - indexComponent
    - indexedComponent
    - positionalComponent
    - requiredComponent

class *Selector*

Selector class

Methods defined here:

__and__(self, other)

Implements the & operator, which returns
clone() refined by other

__call__(self, *args, **kwargs)

Return a new selector consisting of this one refined by the g
Arguments are as per the constructor or method refine.

__init__(self, *args, **kwargs)

Positional args are SelectorComponents or Selectors
Keyword args and their value are passed to kwselect to create
selectors. All the selector components are put into the
components list of this Selector, along with all the componen
of any Selector arguments.

__repr__(self)

clone(self)

```
Makes a copy of this Selector.
```

components(self)

```
List of selector components, each an instance of SelectorComp
```

refine(self, *args, **kwargs)

```
Add components to this selector using the same syntax as the
constructor. Ignores non-keyword arguments that are not
SelectorComponents or Selectors.
```

select(self, variable, *args, **kwargs)

```
Extract the selection from a variable.

May specify additional refinement via extra arguments and
keyword specifiers as in the constructor or method 'refine'.

Options modify the result of the selection. The options and
their default values are:
    -- raw = 0: if 1, return an MA only
    -- squeeze = 0:  If 1, eliminate any dimensions of length
                     from the result.
    -- order = None: If given, is a string such as
                     variable.getOrder()
                     returns. Result is permuted into this or
    -- grid = None:  If given, is a grid object; result is
                     regridded onto this grid.
Each of the components contributes arguments suitable for the
subRegion call in class cdms.AbstractVariable. If a component
is to modify the same axis as a previous component, its appli
is postponed. subRegion is called and the result is then fed
to each of the components' "post" method. This returns a
possibly modified result, which becomes the input to the next
component's post method.

This procedure is repeated until no more components are postp
Then the options are applied to the result in the order
listed above, and the result is returned.

Execption SelectorError is thrown if the selection is
impossible.

The result is a TransientVariable and id(variable) <> id(resu
even if there are no components.
```

unmodified_select(self, variable, raw=0, squeeze=0, order=None, grid=None)

```
Select using this selector without further modification
```

class ***SelectorComponent***

```
Base class representing selection for a given set of axes.
```

Methods defined here:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)

```
Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method.  Set axismap[i] to No
indicate that you have eliminated an axis.
```

specify(self, slab, axes, specifications, confined_by, aux)

```
Refine the specification suitable for slab.subRegion
Set confined_by to yourself for each axis you confine.
If you would normally confine an axis to ':', don't,
unless you *require* that axis not be confined by other
components.

Returning:
   Return 1 if you wish to skip your turn. You'll be called
   later with the results of the other selectors.

   Raise a SelectorError exception if you can't do your job.

   Otherwise, return 0, even if slab had no axes you wish
   to confine.

   Store any info you want in dictionary aux[id(self)]
```

specifyGrid(self, var, grid, specs)

```
Refine the specification suitable for grid.intersect().

'var' is a variable.
'grid' is the grid associated with the variable.
'specs' is the result set of specifications, of the form defi

Return:
  0 if self confines the grid.
  1 if self is not associated with coordinate regions, or doe

Note: This function should return 0 only if self is a compone
  nonrectilinear grids. See class coordinateComponent.
```

class ***SelectorError***(cdms.error.CDMSError)

```
The exception type for errors in the selector packages
```

Method resolution order:

- SelectorError
- cdms.error.CDMSError
- exceptions.Exception

Methods defined here:

__init__(self, args)

Methods inherited from cdms.error.CDMSError:

__str__(self)

Methods inherited from exceptions.Exception:

__getitem__(...)

class ***axisComponent***(SelectorComponent)

```
A SelectorComponent that confines exactly one axis or coordinate di
```

Methods defined here:

__init__(self, id, spec)

__repr__(self)

specify(self, slab, axes, specifications, confined_by, aux)

```
Do specification for axis self.id; skip if axis not present.
```

Methods inherited from SelectorComponent:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)

```
Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method.  Set axismap[i] to No
indicate that you have eliminated an axis.
```

specifyGrid(self, var, grid, specs)

```
Refine the specification suitable for grid.intersect().

'var' is a variable.
'grid' is the grid associated with the variable.
'specs' is the result set of specifications, of the form defi

Return:
  0 if self confines the grid.
```

1 if self is not associated with coordinate regions, or doe

Note: This function should return 0 only if self is a compone
nonrectilinear grids. See class coordinateComponent.

class ***coordinateComponent***(axisComponent)

A SelectorComponent that confines exactly one coordinate dimension

Method resolution order:

coordinateComponent
axisComponent
SelectorComponent

Methods defined here:

__init__(self, id, spec)

specifyGrid(self, var, grid, specs)
Determine if this component confines the grid, and if so set

Methods inherited from axisComponent:

__repr__(self)

specify(self, slab, axes, specifications, confined_by, aux)
Do specification for axis self.***id***; skip if axis not present.

Methods inherited from SelectorComponent:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)
Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method. Set axismap[i] to No
indicate that you have eliminated an axis.

class ***indexComponent***(axisComponent)

An axisComponent that confines exactly one axis by
specifying indices.

Method resolution order:

indexComponent
axisComponent

SelectorComponent

Methods defined here:

__init__(self, id, start=None, stop=None, stride=None)

Methods inherited from axisComponent:

__repr__(self)

specify(self, slab, axes, specifications, confined_by, aux)

Do specification for axis self.***id***; skip if axis not present.

Methods inherited from SelectorComponent:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)

Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method. Set axismap[i] to No
indicate that you have eliminated an axis.

specifyGrid(self, var, grid, specs)

Refine the specification suitable for grid.intersect().

'var' is a variable.
'grid' is the grid associated with the variable.
'specs' is the result set of specifications, of the form defi

Return:
  0 if self confines the grid.
  1 if self is not associated with coordinate regions, or doe

Note: This function should return 0 only if self is a compone
  nonrectilinear grids. See class coordinateComponent.

class ***indexedComponent***(SelectorComponent)

A SelectorComponent that confines exactly one axis
whose index is given.

Methods defined here:

__init__(self, index, value)

specify(self, slab, axes, specifications, confined_by, aux)

Do the specification for axis whose index is self.***index***.

Methods inherited from SelectorComponent:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)

```
Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method.  Set axismap[i] to No
indicate that you have eliminated an axis.
```

specifyGrid(self, var, grid, specs)

```
Refine the specification suitable for grid.intersect().

'var' is a variable.
'grid' is the grid associated with the variable.
'specs' is the result set of specifications, of the form defi

Return:
  0 if self confines the grid.
  1 if self is not associated with coordinate regions, or doe

Note: This function should return 0 only if self is a compone
  nonrectilinear grids. See class coordinateComponent.
```

class ***positionalComponent***(SelectorComponent)

```
A SelectorComponent that confines the next axis available.
```

Methods defined here:

__init__(self, v)

__repr__(self)

specify(self, slab, axes, specifications, confined_by, aux)

```
Find the next unconfined axis and confine it.
```

Methods inherited from SelectorComponent:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)

```
Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method.  Set axismap[i] to No
indicate that you have eliminated an axis.
```

specifyGrid(self, var, grid, specs)

```
Refine the specification suitable for grid.intersect().

'var' is a variable.
'grid' is the grid associated with the variable.
'specs' is the result set of specifications, of the form defi

Return:
  0 if self confines the grid.
  1 if self is not associated with coordinate regions, or doe

Note: This function should return 0 only if self is a compone
  nonrectilinear grids. See class coordinateComponent.
```

class ***requiredComponent***(SelectorComponent)

```
Checks to see that a specific id axis must be present.
```

Methods defined here:

__init__(self, ids)

```
Checks to see that a specific axis or axes must be present.
Initialize with a sequence of ids.
```

specify(self, slab, axes, specifications, confined_by, aux)

```
Doesn't confine but checks for existance.
```

Methods inherited from SelectorComponent:

post(self, fetched, slab, axes, specifications, confined_by, aux, axismap)

```
Post-process fetched if desired, return new value.
Arguments slab, axes, specifications, confined_by, and aux ar
pre-subRegion call.

axismap gives the indices of fetched's axes in axes and shoul
be modified as required by this method.  Set axismap[i] to No
indicate that you have eliminated an axis.
```

specifyGrid(self, var, grid, specs)

```
Refine the specification suitable for grid.intersect().

'var' is a variable.
'grid' is the grid associated with the variable.
'specs' is the result set of specifications, of the form defi

Return:
  0 if self confines the grid.
  1 if self is not associated with coordinate regions, or doe

Note: This function should return 0 only if self is a compone
  nonrectilinear grids. See class coordinateComponent.
```

Functions

kwselect(k, value)

```
Turn a keyword/value pair into a SelectorComponent
The words latitude, longitude, time, and level are
used to pass value to the routine of the same name.
Otherise, axis is called using k as the id.
```

latitude(*value)

```
Creates default selector corresponding to keyword latitude = value
```

latitudeslice(start=None, stop=None, stride=None)

level(*value)

```
Creates default selector corresponding to keyword level = value
```

levelslice(start=None, stop=None, stride=None)

longitude(*value)

```
Creates default selector corresponding to keyword longitude = valu
```

longitudeslice(start=None, stop=None, stride=None)

required(values)

```
Creates a selector that requires a certain axis to be present.
```

setslice(id, start=None, stop=None, stride=None)

time(*value)

```
Creates a default selector corresponding to keyword time=value
```

timeslice(start=None, stop=None, stride=None)

Data

LatitudeType = 'lat'
LongitudeType = 'lon'
TimeType = 'time'
VerticalType = 'lev'
all = Selector()